IV. Thủ tục lưu trữ - 1. Khái niệm

- Thủ tục lưu trữ: là một đối tượng trong CSDL gồm tập nhiều lệnh SQL được nhóm lại thành một nhóm và các lệnh này sẽ được thực hiện khi thủ tục lưu trữ được thực thi.
- Với thủ tục lưu một phần ngôn ngữ lập trình sẽ được đưa vào giúp chúng ta lập trình với cơ sở dữ liệu
- Thủ tục lưu trữ có thể có các thành phần:
  - Cấu trúc điều khiển (IF, WHILE)
  - Biến dùng để lưu các giá trị tính toán, các giá trị truy xuất CSDL
  - Các câu lệnh SQL được kết hợp thành khối lệnh trong thủ tục, một thủ tục có thể có tham số truyền vào hay giá trị trả về giống như ngôn ngữ lập trình thông thường

1

IV. Thủ tục lưu trữ - 2. Ưu điểm

- Đơn giản hóa thao tác dữ liệu do tính module hóa
- Tập trung tại Server nên dễ quản lý
- Được biên dịch một lần và được sử dụng lại kết quả trong lần tiếp theo
- Việc thực thi nhanh hơn so với thực hiện rời rạc các lệnh SQL
- Giảm lưu thông trên mạng do thủ tục lưu cho phép thực hiện bằng câu lệnh đơn giản thay vì nhiều dòng lệnh SQL
- Tăng bảo mật CSDL do việc cấp phát quyền trên thủ tục lưu thay vì tác động trực tiếp đến csdl

2

IV. Thủ tục lưu trữ - 3. Phân loại

- System Stored Procedure:
  - Lưu trữ trong CSDL Master
  - Bắt đầu bằng chữ **sp_***
  - Dùng trong quản trị CSDL và an ninh bảo mật.

  VD: sp_rename –đổi tên bảng/cột…
- Local Stored Procedure:
  - Lưu trong CSDL do người dùng tạo ra
  - Được tạo bởi DBA (Database Administrator) hoặc người lập trình
- Remote Stored Procedure sử dụng thủ tục của một server khác

3

IV. Thủ tục lưu trữ - 3. Phân loại

- Temporary Stored Procedure
  - Có chức năng tương tự Local Stored Procedure
  - Tự hủy khi kết nối tạo ra nó bị ngắt hoặc SQL Server ngừng hoạt động
  - Lưu trên CSDL TempDB
- Extended Stored Procedure:
  - Sử dụng chương trình ngoại vi đã được biên dịch thành DLL.
  - Tên bắt đầu bằng **xp_***

  VD: xp_sendmail dùng gửi mail

  xp_cmdshell dùng thực hiện lệnh của DOS

4

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- RECOMPILE
- Giá trị mặc định
- Giá trị NULL
- Wildcard characters
- Thủ tục có tham số truyền vào
- Thủ tục có giá trị trả về

5

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Tạo thủ tục có lựa chọn RECOMPILE:
- **CREATE {PROC|PROCEDURE } Ten_Thu_Tuc [(danh sách các tham số)] WITH RECOMPILE**

AS

Begin

các câu lệnh của thủ tục

end

VD:

```
CREATE PROC viewlop
WITH RECOMPILE
AS
   SELECT MALOP,TENLOP FROM LOP
```

6

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Tạo thủ tục có giá trị mặc định

```
CREATE {PROC|PROCEDURE } Ten_Thu_Tuc
@ tham số kiểu dữ liệu =giá trị mặc định
AS
   Begin  <các câu lệnh của thủ tục> end
VD:
CREATE PROC view_Lop
   @tenlop nvarchar(10) = '07b1'
AS
   select malop, tenlop
   from lop
   where tenlop=@tenlop
```


IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Tạo thủ tục có giá trị NULL

```
CREATE {PROC|PROCEDURE } Ten_Thu_Tuc
@ tham số kiểu dữ liệu =NULL
AS
   Begin  <các câu lệnh của thủ tục> end
```

Ví dụ: Viết thủ tục hiện danh sách sinh viên có 2 tham số tên lớp nhận NULL và địa chỉ nhận mặc định là Hà Nội

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

```
CREATE PROCEDURE view_sinhvien
@tenlop nvarchar(20) = NULL
@diachi nvarchar(40) = 'Ha Noi'
AS
IF @tenlop IS NULL
   select hoten from sinhvien,lop
   where sinhvien.malop=lop.malop and diachi =@diachi
else
   select hoten from sinhvien,lop where
   sinhvien.malop=lop.malop and diachi =@diachi and
   tenlop=@tenlop
--gọi thủ tục: view_sinhvien; view_sinhvien @tenlop
   =N'07B1'; view_sinhvien @tenlop =N'07B1' @diachi
   =N'Huế'
```


IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Tạo thủ tục sử dụng Wildcard
  - %: chuỗi kí tự bất kì
  - _: 1 kí tự
  - []: kí tự đơn bất kì thuộc giới hạn chỉ định (vd: [a-f] hay [abcdef])
  - [^]: kí tự đơn bất kì không nằm trong giới hạn chỉ định ([^a-f] hay [^abcdef])

```
CREATE  PROCEDURE   viewlop
@tenlop nvarchar(20) = N'A%'
AS
   SELECT * FROM LOP WHERE TENLOP like @tenlop

EXEC viewlop
```


IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Tạo thủ tục có tham số truyền vào
- Xét ví dụ
  - Thêm môn học mới có mã 12 tên là Toán rời rạc và số đơn vị học trình là 4 vào bảng Monhoc
  - Lên điểm môn Toán rời rạc cho các sinh viên của lớp có mã lớp là 1

Như vậy:

1. Insert into monhoc values (12,N'Toán rời rạc',4)
2. Insert into diem (mamon,masv)
   Select 12,masv from sinhvien where malop = 1

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Nhận xét:
  - Khi muốn thêm nhiều môn học mới
  - Khi muốn lên điểm cho các sinh viên ở các lớp khác nhau, nhiều môn khác nhau

⇒Sử dụng thủ tục lưu và có tham số vào

Ví dụ:

Viết thủ tục Lên điểm cho sinh viên cho phép:

+thêm môn học mới với mã môn, tên môn và SoDVHT truyền vào.

+lên điểm các sinh viên của lớp với mã lớp truyền vào.

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

```
Create proc Lendiem
@mamon int, @tenmon nvarchar(30),@sodvht int,
@malop int
AS
Begin
   Insert into mon  values
   (@mamon,@tenmon,@sodvht)
   Insert into diem(mamon,masv)
   Select @mamon,masv from sinhvien
   where malop=@malop
End
exec lendiem 12,'GToan',4,1 -- gọi thủ tục
```

13

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Thủ tục có tham số ra

```
CREATE {PROC|PROCEDURE }  Ten_Thu_Tuc
@ tham số kiểu dữ liệu OUTPUT
AS
   Begin  <các câu lệnh của thủ tục> end
```

- Ví dụ: Viết thủ tục lấy ra tuổi cao nhất của các sinh viên

14

```
Create proc tuoicaonhat
@maxtuoi int output
As
   Select @maxtuoi = max(year(getdate())-
   year(ngaysinh)) from sinhvien

--gọi thủ tục:
   Declare @maxtuoi int =0
   exec tuoicaonhat @maxtuoi output
   select @maxtuoi
```

15

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Thủ tục có chứa biến lưu dữ liệu: biến được khai báo như thông thường
- Ví dụ: Viết thủ tục đưa ra danh sách các sinh viên có tuổi bằng tuổi cao nhất
  - Lấy ra tuổi cao nhất của các sinh viên
  - Chọn các sinh viên có tuổi bằng tuổi cao nhất

16

```
Create proc SV_tuoicaonhat
As
Begin
Declare @maxtuoi int
Select @maxtuoi = max(year(getdate())-
   year(ngaysinh)) from sinhvien
Select hoten, ngaysinh from sinhvien where
year(getdate())-year(ngaysinh) =@maxtuoi
End
```

17

IV. Thủ tục lưu trữ - 4. Tạo thủ tục

- Thủ tục có chứa RETURN trả về giá trị cho đối tượng thực thi thủ tục

```
Create proc tuoicaonhat
As
Begin
Declare @maxtuoi int
Select @maxtuoi = max(year(getdate())-year(ngaysinh))
   from sinhvien
Return @maxtuoi
End
```

```
--gọi thủ tục
Declare @a int
exec @a=tuoicaonhat
select @a
```

18

IV. Thủ tục lưu trữ - 5. Sửa thủ tục

- Cú pháp:

ALTER PROC Tên_thủ_tục [danh sách tham số]
AS
Begin <Tập lệnh> end

- Sử dụng tương tự như CREATE PROC
- Không làm thay đổi quyền được cấp trên thủ tục
- Không tác động đến các thủ tục khác hay trigger phụ thuộc thủ tục này

19

IV. Thủ tục lưu trữ - 6. Xóa thủ tục

- Cú pháp:

DROP PROC Tên_thủ_tục

VD

DROP PROC viewlop

- Lưu ý:
  - Xóa thủ tục thì quyền cấp phát trên thủ tục cũng bị xóa
  - Nếu tạo lại thì phải cấp phát lại quyền

20

7. Thủ tục thêm dữ liệu vào bảng

- ✓ Loại thủ tục này có tham số vào là tên các cột có trong bảng, trừ các cột có kiểu dữ liệu tự tăng (identity)
- ✓ Tham số ra, giá trị trả về: có thể có, cho biết việc thêm dữ liệu có thành công hay không
- ✓ Trong nó chứa câu lệnh T-SQL:
  - Kiểm tra ràng buộc dữ liệu duy nhất (primary key, unique)
    - cú pháp:

*if Exists (Select * from Ten_Bang where Ten_Cot=@ten_cot)*
begin
print 'thông báo lỗi'

21

7. Thủ tục thêm dữ liệu vào bảng

- Kiểm tra ràng buộc khóa ngoại (Foreign Key)
  - Cú pháp

*if not Exists (Select * from Ten_Bang where Ten_Cot=@ten_cot)*
begin
print 'thông báo lỗi"
set @biến_trả_về = giá trị lỗi
return
End

- Kiểm tra ràng buộc Miền giá trị (Check)

if @ten_cot không nằm trong miền giá trị
begin print 'thông báo lỗi" End

22

- Viết thủ tục cho phép thêm lớp:
  - mã lớp, tên lớp là tham số truyền vào
  - mã lớp không bị trùng,
  - nếu trùng thông báo rằng lớp đó đã tồn tại
- Viết thủ tục cho phép thêm sinh viên:
  - Mã sv, họ tên, địa chỉ, mã lớp truyền vào
  - Mã sinh viên không trùng, nếu trùng thông báo
  - Mã lớp phải tồn tại trong bảng Lớp
- Viết thủ tục cho phép nhập điểm sinh viên:
  - Mã sv, mã môn, kết quả truyền vào
  - Mã sinh viên, mã môn phải tồn tại
  - Kết quả nằm trong khoảng từ 0 đến 10

23

```
create proc themlop
@malop int, @tenlop nvarchar(10)
AS
If exists (select * from lop where malop = @malop)
begin print 'ma lop da co' return end
else insert into lop(malop,tenlop) values (@malop,@tenlop)
--gọi thủ tục
exec themlop 11,'07b1'
```

24

```
Create proc themsv
@masv int, @tensv nvarchar(30), @diachi
   nvarchar(40), @malop int
AS
If not exists (select * from lop where malop = @malop)
   print 'ma lop khong ton tai'
else
If exists (select * from sinhvien where masv=
   @masv)print 'ma sinh vien nay da co'
else
begin
insert into sinhvien(masv,tensv,diachi,malop) values
   (@masv,@tensv,@diachi,@malop)
print 'da them thanh cong'
end
```

25

```
create proc nhapdiem
@masv int, @mamon int, @kq int
AS
if exists (select * from diem where mamon =@mamon
   and masv=@masv) print N'Khóa chính bị trùng'
Else If not exists (select * from mon where mamon =
   @mamon) print 'ma mon khong ton tai'
Else If not exists (select * from sinhvien where masv=
   @masv)print 'ma sinh vien nay chưa  co'
Else if (@kq<0) or (@kq>10)print 'Diem khong hợp lệ'
else
Begin insert into diem(masv,mamon,kq) values
   (@masv,@mamon,@kq)
print 'da them thanh cong'
end
```

26

8.Thủ tục cập nhật dữ liệu

- Loại thủ tục này có tham số vào là tên các cột có trong bảng, Tham số ra,giá trị trả về:có thể có, cho biết việc Cập nhật dữ liệu có thành công hay không
- Trong nó chứa câu lệnh T-SQL:
  - Kiểm tra dữ liệu cập nhật có tồn tại hay không
  - Cú pháp:

```
if not Exists (Select * from Ten_Bang where
Ten_Cot_khoa_chinh=@ten_cot_khoa_chinh)
    begin
        print 'thông báo lỗi'
    End
```

27

8.Thủ tục cập nhật dữ liệu

- Cập nhật dữ liệu
- Cú pháp:

```
Update Ten_Bang
Set ten_cot=@tencot [,..]
Where ten_cot_khoa_chinh=@ten_cot_khoa_chinh
```

Chú ý: không cập nhật cột làm khóa chính và cột có thuộc tính identity

28

9.Thủ tục xóa dữ liệu

- Loại thủ tục này có tham số vào là các cột làm khóa chính trong bàng, Tham số ra,giá trị trả về:có thể có, cho biết việc xóa dữ liệu có thành công hay không
- Trong nó chứa câu lệnh T-SQL:
  - Kiểm tra dữ liệu xóa có tồn tại trong bảng nhiều không
  - cú pháp:

```
if Exists (Select * from Ten_Bang_nhieu where
Ten_Cot_khoa_ngoai=@ten_cot_khoa_chinh)
    begin
        print 'thông báo lỗi'
        return
    End
```

29

9.Thủ tục xóa dữ liệu

- Cập nhật dữ liệu
- Cú pháp:

```
Delete from Ten_Bang
Where ten_cot_khoa_chinh=@ten_cot_khoa_chinh
```

  - Chú ý: ngoài các lệnh trên có thể còn các lệnh cập nhật dữ liệu của các bảng liên quan

30

V. Hàm do người dùng định nghĩa

- Hàm Fuction để lấy các giá trị trả về.
- Chia 3 dạng:
  - Hàm trả về giá trị vô hướng (scalar value): giá trị trả về có kiểu là kiểu của SQL như varchar, int, ….
  - Hàm trả về giá trị là bảng tạm (inline table-valued): table
  - Hàm trả về giá trị là một bảng nhưng trong hàm chứa nhiều lệnh

Lưu ý: Hàm khác thủ tục ở chỗ hàm trả về một giá trị thông qua tên hàm, thủ tục thì không.

31

V. Hàm do người dùng định nghĩa

- Hàm trả về giá trị vô hướng:
- **Cú pháp:**

```
Create Function Ten_Ham[(Các tham số)]
Returns Kiểu_dữ_liệu_trả_về
AS
Begin
   <các xử lý>
   Return ket_qua
end
```

32

V. Hàm do người dùng định nghĩa

- Ví dụ: Viết hàm tính tuổi của người có năm sinh là @ns

33

V. Hàm do người dùng định nghĩa

- Ví dụ: Viết hàm tính tuổi của người có năm sinh là @ns

```
Create function fTuoi (@ns int)
Returns int
As
Begin
return year(getdate()) - @ns
end
```

- **Kiểm tra thử hàm**

```
print dbo.fTuoi(1982) --phải có dbo.
```

34

V. Hàm do người dùng định nghĩa

- Hàm trả về giá trị là bảng tạm giống View nhưng có tham số vào
- **Cú pháp:**

```
Create Function Ten_Ham[(Các tham số)]
Returns Table
AS
Begin
   Return(câu lệnh Select)
end
```

35

V. Hàm do người dùng định nghĩa

- Ví dụ: Viết hàm đưa ra danh sách các sinh viên có năm sinh bằng @ns

36

V. Hàm do người dùng định nghĩa

- Ví dụ: Viết hàm đưa ra danh sách các sinh viên có năm sinh bằng @ns

```
Create function fDSach (@ns int)
Returns Table
As
return (select * from SV
        where year(ngaysinh=@ns)
```

- **Kiểm tra thử hàm** (không cần dbo)

```
Select *
From fDSach(1982)
```

37

V. Hàm do người dùng định nghĩa

Hàm gồm nhiều lệnh:

```
Create Function Ten_Ham[(các tham số)]
Returns @Ten_Bang_trả_về Table
(Tên_cột Kiểu_dữ_liệu[,…])
AS
Begin
<các lệnh T-SQL>
Return
End
```

38

V. Hàm do người dùng định nghĩa

- Ví dụ:

Nhanvien(MaNV,TenNV,Diachi)
Donhang(SOHD,MaNV,maKH,ngaydat,ngaygiao, Noigiao)

- Viết hàm đưa ra danh sách các nhân viên có mã nhân viên là tham số truyền vào @manv.
  - Nếu @manv=0 thì đưa ra danh sách tất cả các nhân viên bán được hàng,
  - Ngược lại đưa ra danh danh sách nhân viên bán được hàng có mã bằng mã truyền vào

39

```
Create function ds_nv(@manv int)
returns @bang table
(manv int,
hoten nvarchar(30))
AS
begin
if @manv=0
   insert into @bang
   select a.manhanvien,hoten from nhanvien a , dondathang b
   where a.manhanvien=b.manhanvien
else
   insert into @bang
   select a.manhanvien,hoten from nhanvien a , dondathang b
   where a.manhanvien=b.manhanvien
    and a.manhanvien=@manv
return
end
```

```
--gọi thu tục
select * from ds_nv(2)
```

40

VI. Trigger – 1.Khái niệm

- Trigger là một dạng đặc biệt của thủ tục lưu, chứa các lệnh T-SQL nhằm thực hiện một số hành động nào đó do người lập trình viết
- Khác với thủ tục lưu:
  - Trigger không có tham số
  - Không gọi bằng lệnh EXEC mà tự động kích hoạt khi dữ liệu trên bảng có liên quan đến trigger được cập nhật
- Một Trigger được tạo cho một bảng và dùng để kiểm tra các ràng buộc toàn vẹn phức tạp hoặc cập nhật dữ liệu của các bảng liên quan

41

2.Khi nào dùng Trigger?

- Xét ví dụ:

Khoa(MaK, TenK, SoDT)
Lop(MaLop, TenLop, Siso, MaK)
Sinhvien(MaSV,TenSV,NS,GT,MaLop)
MonHoc(MaMon,TenMon,SoDVHT)
Diem(MaSV,MaMon,Ketqua, Lanthi)

42

2.Khi nào dùng Trigger?

- KHOA có mã khoa là duy nhất
- LOP: Mỗi lớp có 1 mã lớp, có 1 tên lớp và thuộc một khoa. Sĩ số của lớp phụ thuộc vào số sinh viên của lớp đó.
- SINHVIEN: Mỗi sinh viên có 1 mã duy nhất, có 1 tên xác định và thuộc một lớp nào đó.
- MONHOC: mỗi môn có mã môn, tên môn và số đơn vị học trình nhất định.
- DIEM: Cho biết kết quả của sinh viên (MaSV) theo từng môn (MaMon). Kết quả là một số nguyên thuộc khoảng [0,10]
- Với các yêu cầu trên, yêu cầu nào cần phải sử dụng trigger để cài đặt?

43

2.Khi nào dùng Trigger?

- Khi có các ràng buộc mà không thể mô tả khi định nghĩa bảng.
- Khi muốn kiểm soát ràng buộc và đưa ra các thông báo cho người dùng
- Khi có sự thay đổi dữ liệu ở 1 bảng và muốn dữ liệu trên một hay nhiều bảng khác cũng tự động thay đổi theo
- Các xử lý mà muốn tự động thực hiện trên server khi có thao tác thêm, sửa, xóa dữ liệu

44

3. Đặc trưng và hạn chế của Trigger

- Trigger có thể nhận biết, ngăn chặn, hủy bỏ các thao tác làm thay đổi trái phép dữ liệu trong CSDL
- Trigger có thể thực hiện nhiều hành động và có thể được kích hoạt bởi nhiều biến cố (insert, delete, update) nhằm đảm bảo tính hợp lệ của dữ liệu.
- Trigger có thể kiểm tra được các quan hệ phức tạp giữa các bảng mà ràng buộc không thể hiện được.
- Có thể áp dụng trigger cho View
- Trigger không thể tạo trên 1 bảng tạm (tên table có ký tự # hoặc ## phía trước) hoặc bảng hệ thống (system table).
- Trigger loại INSTEAD OF DELETE và INSTEAD OF UPDATE không thể được định nghĩa trên các table có chứa khóa ngoại

45

4. Tạo Trigger

- Cú pháp:

Create Trigger <Ten_trigger> ON Ten_Bang/View

For(After)/Instead of Insert[,Update,Delete]

AS

<Tập lệnh T-SQL>

46

4. Tạo Trigger

- Trong đó:
  - *Tên bảng/view: là bảng/view mà trigger được tạo*
  - *Nếu khai báo với từ khóa For(After) thì trigger sẽ được kích hoạt sau khi dữ liệu đã cập nhật vào bảng*
  - *Nếu khai báo với từ khóa Instead of thì trigger sẽ được kích hoạt trước khi dữ liệu đã cập nhật vào bảng, thường được dùng để kiểm tra dữ liệu cập nhật trên View*
  - *Insert,Update,Delete: Trigger được kích hoạt ứng với biến cố (hành động) thêm, sửa, xóa dữ liệu trong bảng*
- Thứ tự kiểm tra các ràng buộc toàn vẹn dữ liệu trong bảng: Trigger Instead of->constraint-> trigger For

47

4. Tạo Trigger

- Ví dụ: Tạo trigger thông báo khi bảng sinh viên được thêm dữ liệu

```
CREATE TRIGGER thongbao
ON sinhvien
FOR INSERT
AS RAISERROR (N'Bạn đang thêm dữ liệu', 16, 10)
```

48

5. Bảng Inserted và Deleted

- Bảng Inserted và Deleted có cấu trúc tương tự như cấu trúc của bảng mà trigger tác động. Dữ liệu của bảng tùy câu lệnh tác động lên bảng làm kích hoạt Trigger:
- Khi thêm dữ liệu vào bảng, dữ liệu được thêm vào sẽ đưa vào bảng tạm Inserted, khi đó bảng Deleted không có dữ liệu
- Khi xóa dữ liệu khỏi bảng, dữ liệu xóa sẽ đưa vào bảng tạm Deleted, bảng Inserted không có dữ liệu

49

5. Bảng Inserted và Deleted

- Khi thao tác cập nhật (update):
  - ✓ Xóa dòng dữ liệu cũ (dữ liệu cũ sẽ đưa vào bảng Deleted)
  - ✓ Thêm dòng dữ liệu mới (dữ liệu mới sẽ đưa vào bảng inserted)
- Để lấy dữ liệu vừa mới cập nhật vào bảng ta truy cập vào bảng tạm Deleted hoặc Inserted, chỉ truy cập được hai bảng này trong Trigger

50

5. Bảng Inserted và Deleted

- Ví dụ:

```
Create trigger them_khoa
on khoa
after insert
as
select * from inserted
--
insert into khoa(makhoa,tenkhoa)values('T',N'Mỹ thuật')
```

Results | Messages

	makhoa	TenKhoa
1	T	Mỹ thuật

51

Ví dụ

- Ví dụ: Tạo trigger tự động tăng sĩ số sinh viên ở bảng Lớp khi có một sinh viên được thêm vào lớp đó
  - Xác định bảng dữ liệu mà trigger áp dụng
  - Xác định loại trigger instead of/after
  - Xác định biến cố
  - Các xử lý:
    - Kiểm tra mã lớp của sinh viên có tồn tại trong bảng Lớp không?
    - Nếu mã lớp không tồn tại thì thông báo và không cho thêm
    - Ngược lại sẽ tăng sĩ số của lớp lên 1 đơn vị

52

```
CREATE TRIGGER tgIncreaseSiso
ON SinhVien
FOR INSERT
AS
DECLARE @MaLop int
SELECT @MaLop = MaLop FROM INSERTED
IF EXISTS(SELECT * FROM Lop WHERE MaLop=@MaLop)
   BEGIN
   UPDATE Lop SET Siso=Siso + 1
   WHERE MaLop = @MaLop
   END
ELSE
   BEGIN
   RAISERROR('Malop %d chua ton tai', 16,1,@MaLop)
   ROLLBACK TRAN –hủy bản ghi chứa sinh viên vừa thêm
   END
```

53

- Thông báo lỗi trong trigger, cú pháp:

Raiserror('Chuỗi thông báo lỗi',16,1)

- Không cho thay đổi dữ liệu, cú pháp:

Rollback Tran

54

6.Trigger sử dụng mệnh đề IF UPDATE

- Trigger được kích hoạt và thực hiện những thao tác cụ thể khi việc thay đổi dữ liệu chỉ liên quan đến **một số cột nhất định** nào đó của bảng.
- Khi đó sử dụng mệnh đề IF UPDATE trong trigger.
- IF UPDATE không sử dụng được đối với câu lệnh DELETE

55

6.Trigger sử dụng mệnh đề IF UPDATE

- Cú pháp:

```
CREATE TRIGGER tên_trigger
ON tên_bảng
FOR  { [INSERT] [,] [UPDATE]  }
AS
 [IF UPDATE (tên_cột)
 [AND UPDATE (tên_cột) | OR UPDATE (tên_cột)]
    ...]
<các_câu_lệnh_của_trigger >
```

56

6.Trigger sử dụng mệnh đề IF UPDATE

- Ví dụ: Tạo Trigger Update Mamon trên bảng Môn học, nếu mã môn của bảng môn học bị sửa thì trường mã môn của bảng Điểm cũng thay đổi theo.
-

57

```
CREATE TRIGGER tgUpdateMaMon
ON mon
FOR UPDATE
AS
IF UPDATE(MaMon)--NEU COT MAMON BI SUA DOI
BEGIN
DECLARE @MaMonCu int,
        @MaMonMoi int
SELECT @MaMonCu = MaMon FROM DELETED
SELECT @MaMonMoi = MaMon FROM INSERTED
UPDATE Diem SET MaMon=@MaMonMoi
WHERE MaMon= @MaMonCu
END
```

58

- Kiểm tra:

```
Update Mon SET MaMon = 1
WHERE MaMon=12
```

- Câu lệnh trên sẽ kích hoạt Trigger vì cột Mã môn được update
- Nhưng câu lệnh sau sẽ không kích hoạt trigger

```
Update Mon
Set SoDVHT=4
Where Mamon = 1
```

59

7. Sửa, xóa Trigger

- Sửa: ALTER TRIGGER <tên_trigger>
- Xóa: DROP TRIGGER <tên_trigger>
- Làm mất hiệu lực của trigger:

ALTER TABLE <tên_bảng> DISABLE TRIGGER <tên_trigger>

- Làm trigger có hiệu lực:

ALTER TABLE <tên_bảng> ENABLE TRIGGER <tên_trigger>/[ALL]

60

Bài tập

- Tạo trigger cho phép giảm sĩ số của lớp khi có một sinh viên bị xóa.
- Viết trigger để kiểm tra mã sv có tồn tại trong bảng Sinh viên khi thực hiện thêm dữ liệu vào bảng Điểm.
- Viết trigger kiểm soát việc thêm dữ liệu vào bảng Diem với trường Kết quả có thỏa mãn nằm trong khoảng [0,10]
- Viết trigger xóa lớp: nếu lớp không có sinh viên thì cho phép xóa, ngược lại thông báo không được xóa.

61

Khi thêm mới mẫu tin :

- Trigger của sự kiện này sẽ tự động kích hoạt khi có một bản ghi được thêm vào bảng dữ liệu. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như:
  - Khóa ngoại.
  - Miền giá trị.
  - Các thuộc tính liên quan trong cùng một bảng.
  - Các thuộc tính liên quan của nhiều bảng khác nhau.

62

Khi hủy bỏ mẫu tin :

- Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng bị xóa. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là :
  - Khóa ngoại để xóa tự động các dữ liệu bên bảng nhiều có liên quan hoặc thông báo cho người dùng

63

Khi sửa mẫu tin :

- Trigger của sự kiện này sẽ tự động kích hoạt khi dữ liệu trong bảng bị sửa đổi. Thông thường bên trong trigger sẽ có một số các kiểm tra ràng buộc toàn vẹn dữ liệu như là :
  - Khóa ngoại.
  - Miền giá trị.
  - Các thuộc tính liên quan trong cùng một bảng.
  - Các thuộc tính liên quan của nhiều bảng khác nhau.
  - Thường chỉ cho phép sửa đổi trên một số cột nhất định.

64

8. Tranaction – Giao tác

- Giao tác là các hành động cập nhật dữ liệu trên nhiều bảng khác nhau được thực hiện trong cùng một khối.
- Ví dụ: Một khách hàng có cùng lúc 2 tài khoản trong ngân hàng: thanh toán và tiết kiệm.
  - Tài khoản thanh toán dùng để thực hiện các giao dịch nộp hoặc chuyển khoản của khách hàng với đối tác
  - Tài khoản tiết kiệm dùng để gửi tiền tiết kiệm lấy lãi cuối kì theo kì hạn 3 tháng

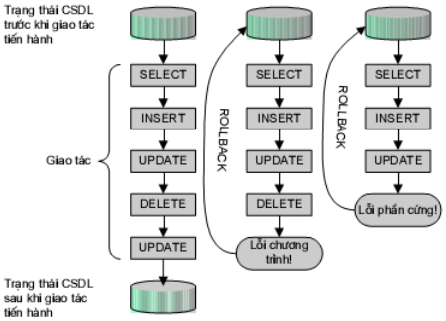
65

8. Tranaction – Giao tác

- BEGIN TRANSACTION<tên giao tác>: Bắt đầu một giao tác
- SAVE TRANSACTION <tên điểm đánh dấu>: Đánh dấu một vị trí trong giao tác (gọi là điểm đánh dấu).
- ROLLBACK TRANSACTION <tên giao tác>: Quay lui trở lại đầu giao tác hoặc một điểm đánh dấu nào đó trong giao tác.
- COMMIT TRANSACTION <tên giao tác>: Đánh dấu điểm kết thúc một giao tác.
- ROLLBACK [WORK]: Quay lui trở lại đầu giao tác.
- COMMIT [WORK]: Đánh dấu kết thúc giao tác.

66

8. Tranaction – Giao tác

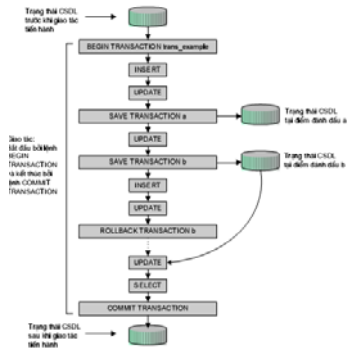


8. Tranaction – Giao tác

Ví dụ 1:
BEGIN TRANSACTION giaotac1
UPDATE monhoc SET sodvht=4 WHERE sodvht=3
UPDATE lop SET malop=2 WHERE tenlop = '07B1'
ROLLBACK TRANSACTION giaotac1

Ví dụ 2
BEGIN TRANSACTION giaotac1
UPDATE monhoc SET sodvht=4 WHERE sodvht=3
UPDATE lop SET malop=2 WHERE tenlop = '07B1'
COMMIT TRANSACTION giaotac1

8. Tranaction – Giao tác



8. Tranaction – Giao tác

Ví dụ 3:
BEGIN TRANSACTION giaotac3
UPDATE monhoc SET sodvht=4 WHERE sodvht=3
SAVE TRANSACTION a
UPDATE lop SET malop=2 WHERE tenlop='07B1'
ROLLBACK TRANSACTION giaotac3
UPDATE monhoc SET sodvht=3 WHERE sodvht=2
COMMIT TRANSACTION giaotac3